TP2 – Erreur de généralisation et validation croisée

Il est nécessaire d'avoir terminé les trois premiers exercices du TP1 avant de commencer le TP2. Dans le TP1, le degré d de la courbe de Bézier n'a pas été considéré comme une inconnue. Le TP2 vise à présenter deux techniques permettant d'estimer ce degré.

Données de test

S'il est possible de générer n_{test} nouvelles données censées correspondre au degré d recherché, ces données $\mathcal{D}_{\text{test}} = \{(x_k, y_k), k = 1 \dots n_{\text{test}}\}$ sont appelées « données de test », à ne pas confondre avec les données d'apprentissage $\mathcal{D}_{\text{app}} = \{(x_j, y_j), j = 1 \dots n_{\text{app}}\}$. En général, on choisit n_{test} grand, car le risque empirique calculé sur $\mathcal{D}_{\text{test}}$ est un estimateur non biaisé de l'erreur de généralisation.

Le script `exercice_0`, qui est très similaire au script `exercice_1` du TP1, permet de produire $n_{\text{test}} = 200$ données de test (x_k, y_k) d'abscisses uniformément réparties dans l'intervalle $[0, 1]$, avec les mêmes paramètres que ceux utilisés pour la création des données d'apprentissage. Après avoir recopié dans le répertoire ad hoc les fonctions déjà écrites pour le TP1, vérifiez que le script `exercice_0` produit correctement les données de test.

Exercice 1 : calcul de l'erreur de généralisation

L'erreur de généralisation (ou *risque espéré*) est définie comme l'écart quadratique moyen entre les données de test y_k et les prédictions obtenues à partir du modèle appris $f(\beta_0^*, \widehat{\beta}, \beta_5^*, x_k)$, où $\widehat{\beta}$ est estimé à partir des données d'apprentissage (et non pas à partir des données de test !).

Écrivez la fonction `erreur_generalisation`, appelée par le script `exercice_1`, qui calcule l'erreur de généralisation pour la valeur du degré d passée en paramètre.

Écrivez la fonction `estimation_1_d_sigma`, appelée par le script `exercice_1`, qui doit retourner : une estimation $\widehat{d}$ du degré d de la courbe de Bézier égale au minimiseur de l'erreur de généralisation ; une estimation $\widehat{\sigma}$ de l'écart-type σ du bruit sur les données égale à la racine carrée du minimum de l'erreur de généralisation.

Exercice 2 : validation croisée de type *Leave-one-out*

S'il n'est pas possible de générer des données supplémentaires, on peut se servir des données d'apprentissage $\mathcal{D}_{\text{app}}$ pour tester le modèle. Cette approche s'appelle la « validation croisée » (*cross validation*). Dans sa version *Leave-one-out*, qui signifie « une [donnée] laissée de côté », la validation croisée VC, qui dépend du degré d de la courbe de Bézier, s'écrit :

$$VC = \frac{1}{n_{\text{app}}} \sum_{j=1}^{n_{\text{app}}} \Big[y_j - f(\beta_0^*, \widehat{\beta}_j, \beta_5^*, x_j) \Big]^2 \tag{1}$$

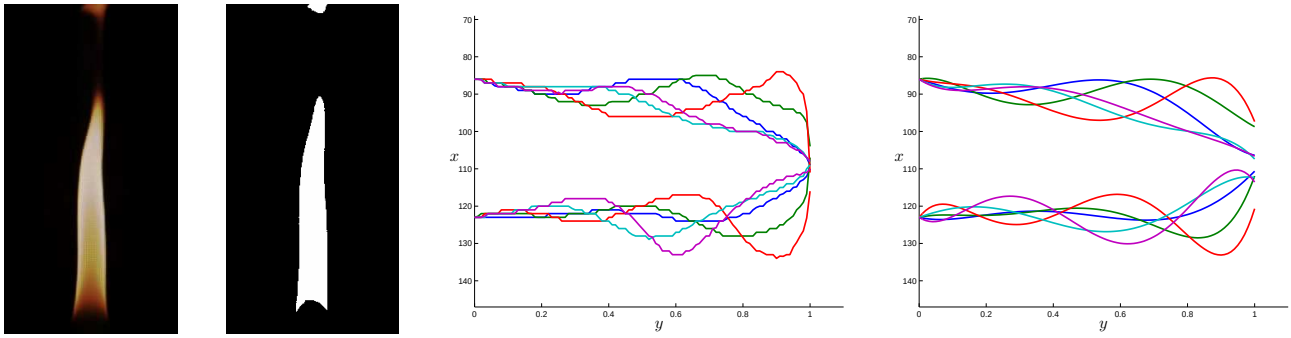
où chaque vecteur de paramètres $\widehat{\beta}_j$ est estimé avec les données d'apprentissage $\mathcal{D}_{\text{app}} \backslash \{(x_j, y_j)\}$. Cela signifie que chaque point (x_j, y_j) sert $n_{\text{app}} - 1$ fois de donnée d'apprentissage, et une fois de donnée de test.

Écrivez la fonction `calcul_VC`, appelée par le script `exercice_2`, qui doit calculer l'expression (1) de VC pour la valeur du degré d passée en paramètre.

Écrivez la fonction `estimation_2_d_sigma`, appelée par le script `exercice_2`, qui doit retourner : une nouvelle estimation $\widehat{d}$ du degré d de la courbe de Bézier égale au minimiseur de VC ; une nouvelle estimation $\widehat{\sigma}$ de l'écart-type σ du bruit sur les données égale à la racine carrée du minimum de VC.

Application : simulation d'une flamme de bougie (partie facultative)

On souhaite simuler, de la façon la plus réaliste possible, une séquence d'images d'une flamme de bougie. La figure 1 illustre les prétraitements effectués sur une séquence réelle de n images : chacune de ces images (cf. figure 1-a), soumise à un seuillage, fournit une image binaire (cf. figure 1-b), sur laquelle la silhouette de la flamme est détectée. Après normalisation, toutes ces silhouettes ont une même hauteur égale à 1. En tournant les axes d'un quart de tour vers la droite, les abscisses x sont orientées vers le bas et les ordonnées y vers la droite (cf. figure 1-c). Lancez le script `donnees`, afin de créer la matrice tridimensionnelle `bords`, de taille $m \times 2 \times n$, où $m = 101$ désigne le nombre de lignes correspondant à la hauteur de la flamme, et où $n = 10$ désigne le nombre d'images de la séquence, de telle sorte que `bords(p,1,q)` et `bords(p,2,q)`, pour $p = 1 \dots m$ et $q = 1 \dots n$, contiennent les abscisses des bords gauche et droit de la $q^{\text{ème}}$ silhouette (devenus ses bords supérieur et inférieur, après rotation d'un quart de tour), à l'ordonnée $y = (p-1)/(m-1)$. La figure 1-c montre que les silhouettes ont toutes la même base, c'est-à-dire que `bords(1,1,q)` et `bords(1,2,q)` ne dépendent pas de q (et valent, respectivement, 86 et 123).



(a) (b) (c) (d)

FIGURE 1 – (a) Une des images de la séquence réelle. (b) Détection de la silhouette par seuillage. (c) Silhouettes de la séquence réelle, après normalisation et rotation. (d) Modélisation des silhouettes par des paires de courbes de Bézier indépendantes.

Chaque bord de chaque silhouette peut être modélisé par une courbe de Bézier de degré d, entièrement définie par $d+1$ points de contrôle, dont les ordonnées $\alpha_i = i/d$, $i = 0 \dots d$, sont équiréparties dans l'intervalle $[0, 1]$. Les points de contrôle sont notés $P_i^q = (\beta_i^q, \alpha_i)$ pour le bord gauche (supérieur) de la $q^{\text{ème}}$ silhouette, et $Q_i^q = (\gamma_i^q, \alpha_i)$ pour son bord droit (inférieur). Les abscisses $\beta_0^q = 86$ et $\gamma_0^q = 123$ étant indépendantes de q, sont notées β_0 et γ_0. Les bords de la silhouette numéro q sont donc modélisés par les équations :

$$\begin{cases} x = \beta_0 \, B_0^d(y) + \displaystyle\sum_{i=1}^{d} \beta_i^q \, B_i^d(y) \\ x = \gamma_0 \, B_0^d(y) + \displaystyle\sum_{i=1}^{d} \gamma_i^q \, B_i^d(y) \end{cases}$$

où $y \in [0, 1]$ et où B_i^d désigne le polynôme de Bernstein de degré d.

Manifestement, modéliser la silhouette numéro q consiste à estimer les d paramètres $\beta^q = [\beta_1^q, \dots, \beta_d^q]$ de son bord gauche et les d paramètres $\gamma^q = [\gamma_1^q, \dots, \gamma_d^q]$ de son bord droit. La figure 1-d montre les modélisations obtenues à partir des silhouettes de la figure 1-c. Il est notable que les sommets des flammes ne sont pas fermés.

Exercice 3 : modélisation par deux courbes de Bézier couplées

Pourque les deux courbes de Bézier se rejoignent au sommet de la flamme, il faut les coupler, c'est-à-dire faire en sorte que le point de contrôle situé au sommet de la flamme, c'est-à-dire à l'ordonnée $y = 1$, soit commun aux deux courbes. Il faut donc reformuler le problème sous la forme d'un nouveau système linéaire :

$$\mathbf{C}^q \, \mathbf{X}^q = \mathbf{D}^q \tag{2}$$

où le vecteur des inconnues vaut $\mathbf{X}^q = [\beta_1^q, \ldots, \beta_{d-1}^q, \gamma_1^q, \ldots, \gamma_d^q]^\top$, sachant que β_d^q ne fait pas explicitement partie des inconnues puisque $\beta_d^q = \gamma_d^q$.

Établissez (sur papier) les expressions de la matrice $\mathbf{C}^q$ et du vecteur $\mathbf{D}^q$ de l'équation (2). Attention : cette étape demande à être traitée avec soin.

Écrivez la fonction `moindres_carres_bis`, appelée par le script `exercice_3`, permettant de résoudre le système linéaire (2) au sens des moindres carrés ordinaires.

Exercice 4 : simulation de silhouettes par tirages aléatoires

L'analyse précédente permet de caractériser la silhouette d'une flamme par $2\,d - 1$ paramètres. Pour simuler une silhouette,on modélise les distributions des abscisses des points de contrôle par des lois normales, puis on utilise les lois estimées pour procéder au tirage aléatoire de nouveaux points de contrôle (fonction `randn`).

Écrivez la fonction `estimation_lois_n`, appelée par le script `exercice_4`, permettant d'estimer la moyenne et l'écart-type de chacun des points de contrôle, considéré comme une variable aléatoire, à partir des réalisations empiriques que constituent les données réelles.

Écrivez la fonction `simulation`, appelée par le script `exercice_4`, qui doit simuler une silhouette de flamme par tirage aléatoire de valeurs pour les paramètres $[\beta_1^q, \ldots, \beta_{d-1}^q, \gamma_1^q, \ldots, \gamma_d^q]$, en utilisant les paramètres de lois normales précédemment estimés.

En guise de conclusion de cette partie facultative, le script `sequence_flammes` montre une application de l'estimation des paramètres de courbes de Bézier à la « réalité augmentée ».